Beutnagel 4-1-13-3

#11 Dec/affidavit
5/1/03

IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

Patent Application

Inventor(s)	Mark Beutnagel Ariel Fischer Joern Ostermann Yao Wang	Case Name	Beutnagel 4-1-13-3
Filing Date	12/31/1998	Serial No.	09/224,583
Examiner	Michael Opsasnick	Group Art Unit	2654
Title	Integration of Talking Heads and Text-to-Speech Synthesizers for Visual TTS		

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON, D.C. 20231
SIR:

DECLARATION UNDER 37 CFR 1.131

1. With reference to US Patent 6,181,351, which was filed on April 13, 1998, I hereby declare the following:
2. My co-inventors and I have invented the subject matter claimed in the instant application prior to April 13, 1998.
3. In support of this assertion, enclosed is a photocopy of a letter and an accompanying memorandum that was sent by a Vice President of AT&T Labs, Dr. L. Rabiner, to the IP Department, asking that a patent application be prepared. This letter is dated prior to April 13, 1998.

Respectfully,

Dated: 4/20/03

Joern Ostermann

AT&T

AT&T Labs - Research

Subject: Patent Application
Work Project No. 311615-5006

date:

from: L. R. Rabiner
FP-D151
973-360-8500

Tom Restaino:
LC Suite 3000

Attached for patent consideration is a memo entitled "FAP definition syntax for TTS input," by Ariel Fischer, Yao Wang, Mark Beutnagel and Joern Ostermann of AT&T Labs - Research.

This document describes the syntax used to define FAP bookmark sequences as input to a TTS system. The purpose of adding this functionality is to allow the control of facial animations (smile, sadness, ...) directly from the input text of the TTS. For this kind of animations, simply applying an FAP of a constant value and removing it after a certain amount of time does not give a realistic face motion. The proposal allows the user to design complex timing behavior, and thus to have a high level of freedom for defining the evolution of the FAP amplitude.

Thank you for your consideration. If you have any further questions, please contact Joern Ostermann on Ext. 3311.

L. R. Rabiner

NSL-HA6155000/JO/sef
Att.

Copy to
Mark Beutnagel
Ariel Fischer
B. G. Haskell
J. Ostermann
Yao Wang

Ariel Fischer, Yao Wa ark Beutnagel, Jörn Ostermann
AT&T Labs - Research

FAP definition syntax for TTS input

This document describes the syntax used to define FAP bookmark sequences as input to a TTS system. The purpose of adding this functionality is to allow the control of facial animations (smile, sadness, ...) directly from the input text of the TTS. For this kind of animations, simply applying an FAP of a constant value and removing it after a certain amount of time does not give a realistic face motion. The proposal allows the user to design complex timing behavior, and thus to have a high level of freedom for defining the evolution of the FAP amplitude.

The following figure shows the complete blockdiagram describing the integration of a proprietary TTS Synthesizer into an MPEG-4 face animation system. The FAP bookmarks defined by the user in the input text of the TTS Stream are identified by the speech synthesizer and transmitted in ASCII format to the Phoneme to FAP converter.

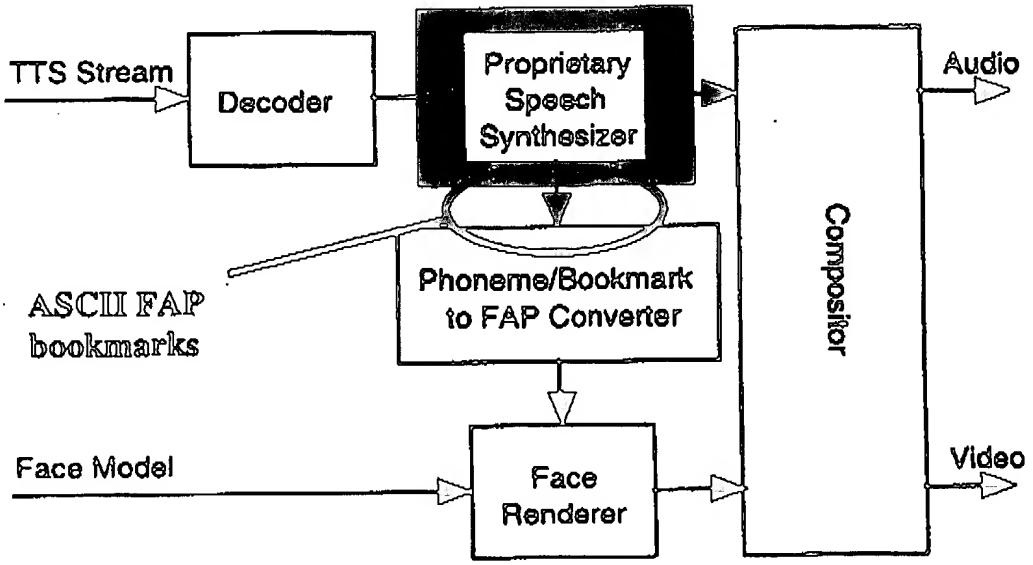

Figure 1: Blockdiagram showing the integration of a proprietary TTS into an MPEG-4 face animation system

The syntax of the bookmark sequences used to convey commands to the TTS system is the following, repeated as many times as the user wants:
<FAP # (FAPselect) FAPval FAPdur>
#: defined according to the visualFCD, Annex C, Table 12-1
FAPselect: defined according to Table 12-3 (expression select), in case # == 2
FAPselect: defined according to Table 12-5 (viseme select), in case # == 1
FAPval: defined in units according to Table 12-1
FAPdur: defined in ms

The Phoneme/Bookmark to FAP Converter (Fig. 1) is responsible for translating the FAP bookmarks defined by the user into an FAP stream that can be interpreted by the Face Renderer.

Ariel Fischer, Yao W[illegible]iark Beutnagel, Jörn Ostermann
AT&T Labs - Research

FAPval defines the value of the FAP to be applied at the end of FAPdur. The value of the FAP at the beginning of the action (startValue) depends on the previous value and can be equal to:

- 0 if the FAP bookmark sequence is the first one with this FAP #
- FAPval of the previous defined FAP with the same FAP # if a time longer than the previous FAPdur has elapsed between the two FAP definitions.
- The actual reached value due to the previous FAP definition if a time shorter than the previous FAPdur has elapsed between the two FAP definitions.

At the end of FAPdur, FAPval is maintained until another FAP definition gives a new value to reach. To reset the action, an FAP with FAPval equal to 0 must be defined.

To avoid too many parameters for defining the evolution of the value between the beginning of the action and its end, the function that computes for each frame the value of the FAP to be sent to the face animation system is predefined. We implemented the following functions:

$$(1): linear \quad / \quad (2): 1-e^{-t} \quad / \quad (3): \frac{1}{1-e^{-\lambda(t-\frac{FAPdur}{2})}} \quad / \quad (4): Hermite$$

All these function uses as input the starting value (determined as explained before), FAPval and FAPdur, and thus is completely determined as soon as the FAP definition is known. After extensive subjective evaluations, it turns out that the Hermite function of third order gives the best results, in terms of realistic behavior. Using Splines with more than one Hermite segment would increase the flexibility for designing curves but would also require to have some knowledge with regards to values placed further in the text than the FAP definition, which is a significant drawback for a real-time system.
The Hermite function of third order enables one to match the tangent at the beginning of a segment with the tangent at the end of the previous segment, so that a smooth curve can be guarantied.
The computation of the Hermite function requires 4 parameters as input, which are: the value of the first point of the curve (startValue), its tangent (startTangent), the value to be reached at the end of the curve (equal to FAPVal) and its tangent (always set to 0 in our implementation).

For each FAP#, the first curve (due to FAP# $bookmark_{i=0}$) has a startValue ($startValue_{i=0}$) equal to 0 and a startTangent ($startTangent_{i=0}$) also equal to 0. The value for startTangent and startValue for $i > 0$ depends on the time elapsed between FAP# $bookmark_{i-1}$ and FAP# $bookmark_i$ ($t_{i-1;i}$).
If $t_{i-1;i} > FAPdur_{i-1}$ then:
$startValue_i = FAPval_{i-1}$
$startTangent_i = 0$
and the resulting amplitude of the FAP to be sent to the renderer is computed with equation (4.1):

$$(4.1): \; FAPAmpl_i(t) = startValue_i \cdot (2t^3 - 3t^2 + 1) + FAPval_i \cdot (-2t^3 + 3t^2) + startTangent_i \cdot (t^3 - 2t^2 + t)$$
$$with \; t \in [0 \quad 1]$$

$FAPdur_i$ is used to relocate and scale the time parameter, t, from [0 1] to [t_i t_i+$FAPdur_i$] with t_i being the instant when the word following FAP# $bookmark_i$ in the text is pronounced. Thus, equation (4.2) gives the exact rendering time:

$$(4.2): Rendering\ time\ for\ FAPAmpl_i(t) = t_i + t * FAPdur_i$$

Ariel Fischer, Yao Wa[illegible]lark Beutnagel, Jörn Ostermann
AT&T Labs - Research

If $t_{4;1;1} < FAPdur_{4;1}$ then:
$startValue_4 = FAPAmpl_{4;1}(t_{4;1;1} / FAPdur_{4;1})$
$startTangent_4 = Tangent_{4;1}(t_{4;1;1} / FAPdur_{4;1})$ which is computed with equation (4.3):

$$(4.3): \underset{i-1}{Tangent}(t) = \underset{i-1}{startValue} \cdot (6t^2 - 6t) + \underset{i-1}{FAPval} \cdot (-6t^2 + 6t) + \underset{i-1}{startTangent} \cdot (3t^2 - 4t + 1)$$
$$with \ t \in [0 \quad 1]$$

and the resulting amplitude of the FAP is again computed with equation (4.1).

The next figure shows an example of a timing curve created with 3 bookmarks sequences for FAP 2 (expression) and FAPSelect 1 (joy).

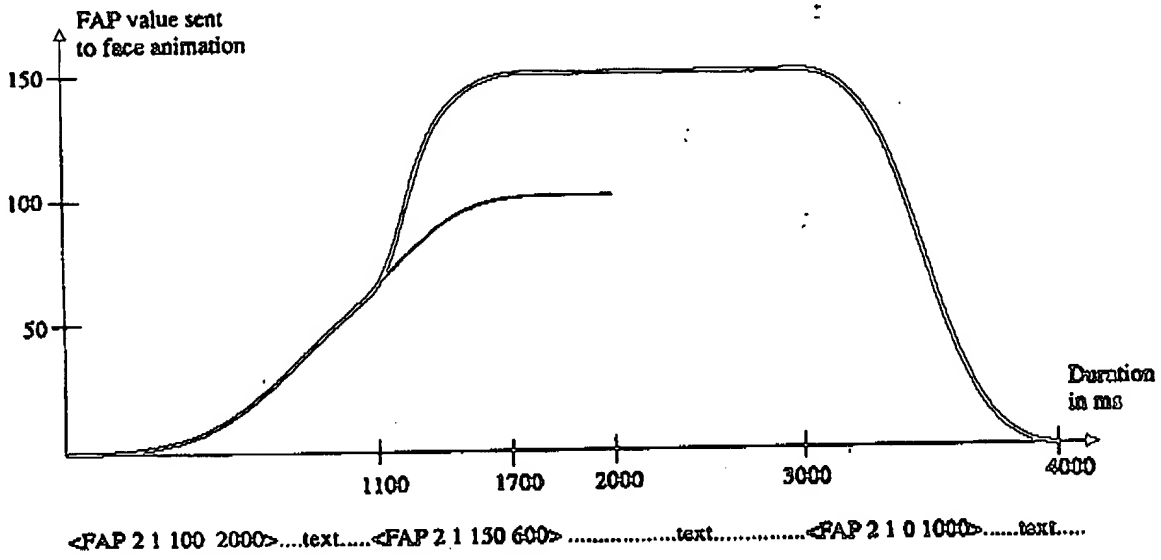

Figure 2: Example of a timing curve